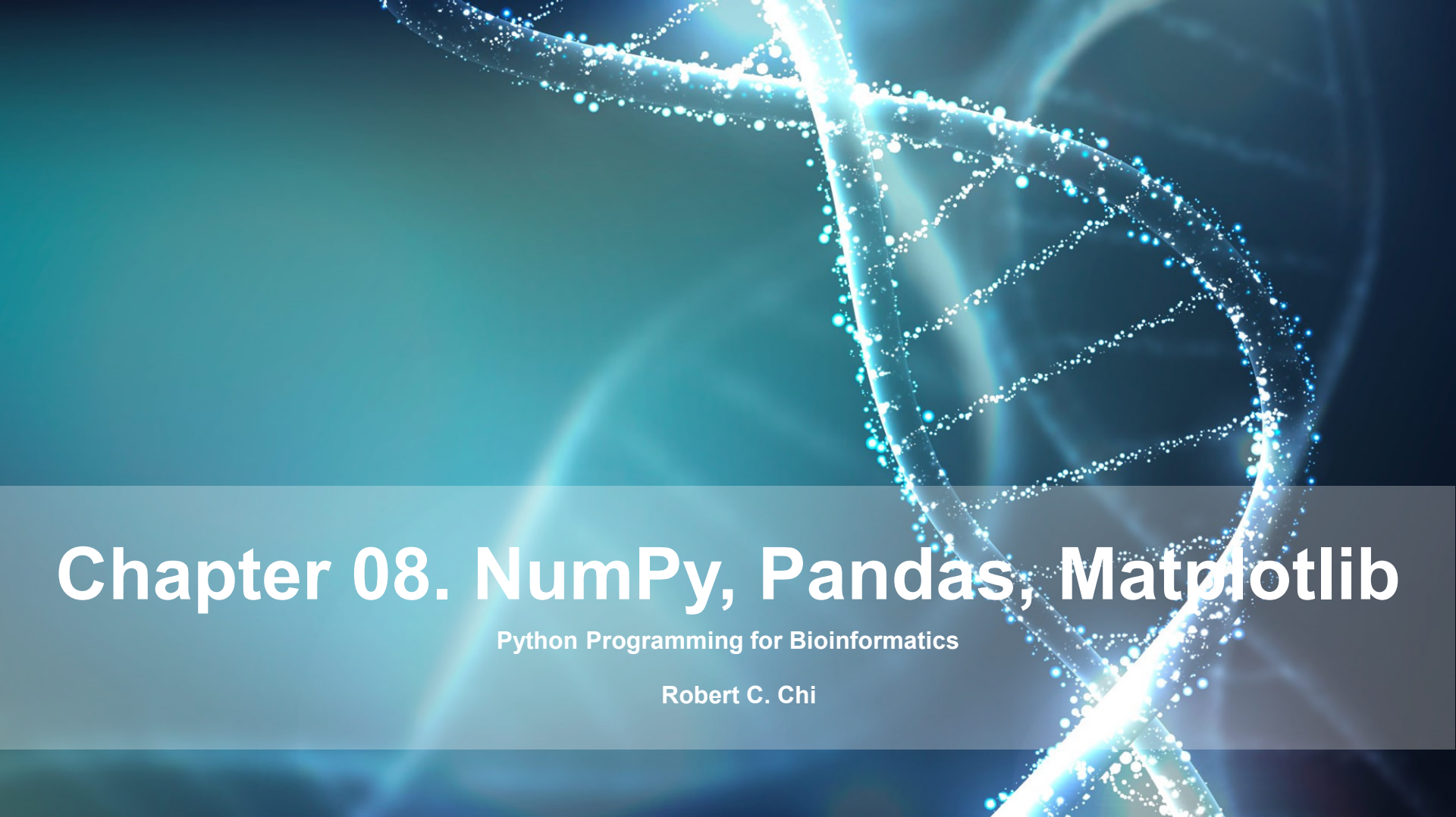


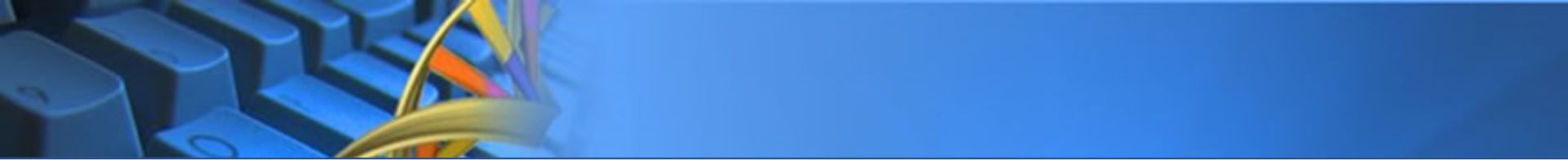
Chapter 08. NumPy, Pandas, Matplotlib

Python Programming for Bioinformatics

Robert C. Chi

Agenda

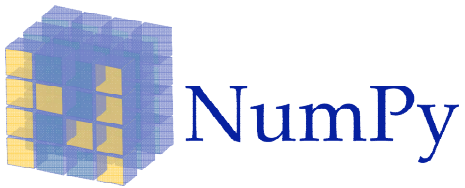
- NumPy
- Pandas
- MatPlotLib



NUMPY

NumPy 簡介

- **The role of NumPy**
  - Handling "**Arrays**"
  - "**Array**" = A collection of data with same type
  - It is the **base framework** for pandas, matplotlib...etc.
- **Installation**
  - Use the following commands in Google Colab:
    - !pip install numpy
- **Import**
  - import numpy as np



Create an Array

- **Create a one-dimensional array**

```
1  import numpy as np
2
3  ary1 = np.array([1, 2, 3])
4  print(ary1)
```

```
[1 2 3]
```

- **Create a two-dimensional array**

```
1  import numpy as np
2
3  ary2 = np.array([[1, 2, 3], [4, 5, 6]])
4  print(ary2)
```

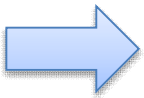
```
[[1 2 3]
 [4 5 6]]
```

Practice

- **Create an Array**
  - Please create arrays with the following codes:
    - ary1 = np.array([1, 2, 3])
    - ary2 = np.array([[1, 2, 3], [4, 5, 6]])

Calculation of Statistics

```
1  import numpy as np
2
3  stat1 = np.arange(1, 11)
4  print(stat1)
5
6  print(stat1.min())
7  print(stat1.max())
8  print(stat1.sum())
9  print(stat1.mean())
10 print(stat1.std())
```



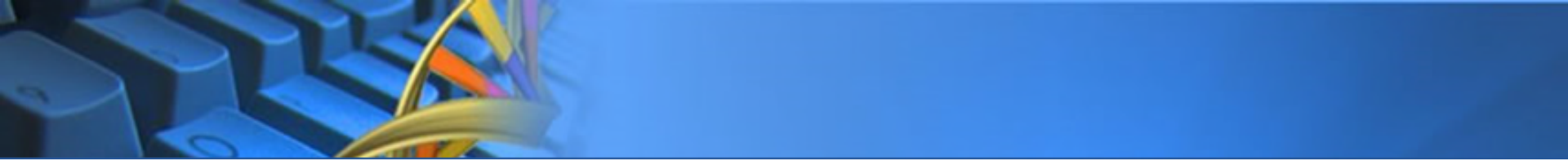
```
[ 1  2  3  4  5  6  7  8  9 10]
1
10
55
5.5
2.8722813232690143
```

1. Original Data
2. Minimum in the Array
3. Maximum in the Array
4. Sum of the Array
5. Average of the Array
6. Standard deviation of the array

Practice

- **Calculation of Statistics**
  - Please create the following array first:
    - stat1 = np.**arange**(1, 11)
  - Please calculate the statistics of the array with following codes:
    - **Minimum:** stat1.**min**()
    - **Maximum:** stat1.**max**()
    - **Sum:** stat1.**sum**()
    - **Average:** stat1.**mean**()
    - **Standard Deviation:** stat1.**std**()
  - Codes for reference:

```
1   import numpy as np
2
3   stat1 = np.arange(1, 11)
4   print(stat1)
5
6   print(stat1.min())
7   print(stat1.max())
8   print(stat1.sum())
9   print(stat1.mean())
10  print(stat1.std())
```



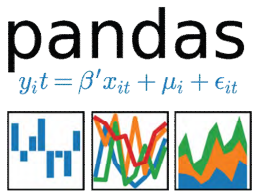
PANDAS

Introduction of Pandas

- **The role of Pandas**
  - Read the "External Data" and present it in "Table".
  - The Data Type of 2D Table: DataFrame

- **Installation**
  - Use the following commands in Google Colab:
    - !pip install pandas

- **Import**
  - import pandas as pd



Create a DataFrame

- 透過「**CSV 檔**」建立

CarSales.csv

	A	B	C	D
1	Country	Age	Salary	ToBuy
2	France	44	72000	No
3	Spain	27	48000	Yes
4	Germany	30	54000	No
5	Spain	38	61000	No
6	Germany	40		Yes
7	France	35	58000	Yes
8	Spain		52000	No
9	France	48	79000	Yes
10	Germany	50	83000	No
11	France	37	67000	Yes

```
1  import pandas as pd
2
3  dfCSV = pd.read_csv("CarSales.csv")
4  print(dfCSV)
5
6  print(dfCSV["Country"].mode())
7  print(dfCSV["Age"].median())
8  print(dfCSV["Salary"].mean())
```

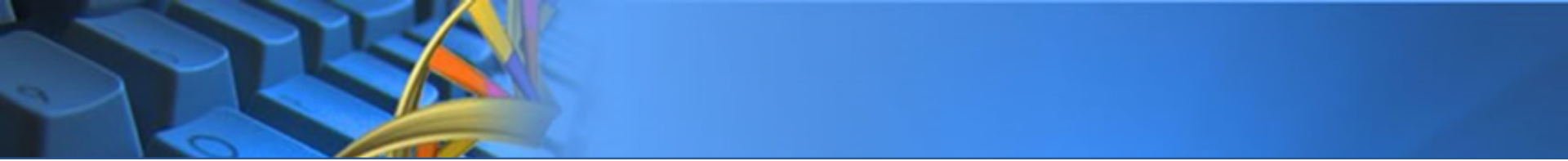
```
   Country   Age   Salary ToBuy
0   France  44.0  72000.0    No
1    Spain  27.0  48000.0   Yes
2  Germany  30.0  54000.0    No
3    Spain  38.0  61000.0    No
4  Germany  40.0      NaN   Yes
5   France  35.0  58000.0   Yes
6    Spain   NaN  52000.0    No
7   France  48.0  79000.0   Yes
8  Germany  50.0  83000.0    No
9   France  37.0  67000.0   Yes
```

- **NaN** = Not a Number
- The Symbol for missing data

Practice

- **Create a DataFrame**
  - Test the following codes in Google Colab:

```
1  import pandas as pd
2
3  dfCSV = pd.read_csv("CarSales.csv")
4  print(dfCSV)
5
6  print(dfCSV["Country"].mode())
7  print(dfCSV["Age"].median())
8  print(dfCSV["Salary"].mean())
```



MATPLOTLIB

Matplotlib.PyPlot

- **The role of Matplotlib.pyplot**
  - Draw various "statistical charts"
  - Visualize the data

- **Installation**
  - Use the following commands in Google Colab:
    - !pip install matplotlib

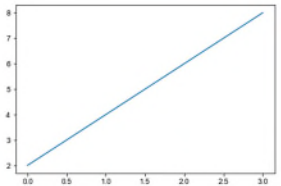
- **Import**
  - import matplotlib.pyplot as plt

Line Chart: plot()

- **Syntax**
  - plt.plot([X-Data], Y-Data, [Line-Style] …)

- **Only provide Y-Data:**

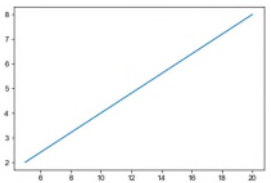
```
1  import matplotlib.pyplot as plt
2
3  plt.plot([2, 4, 6, 8])
4  plt.show()
```



Default of X :
[0, 1, 2, 3…]

- **Provide X-Data & Y-Data**

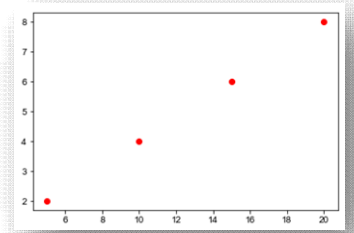
```
1  import matplotlib.pyplot as plt
2
3  plt.plot([5, 10, 15, 20], [2, 4, 6, 8])
4  plt.show()
```



Line Chart: plot()

- **Provide X, Y Data and Line Styles**

```
1  import matplotlib.pyplot as plt
2
3  plt.plot([5, 10, 15, 20], [2, 4, 6, 8], "ro")
4  plt.show()
```



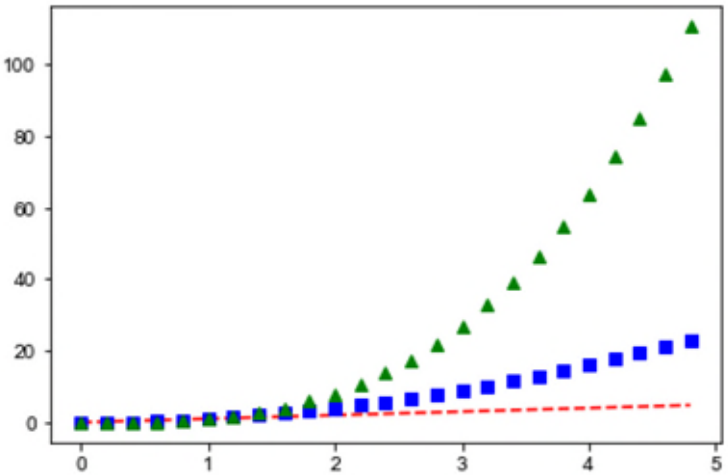
Sym.	Color
"r"	Red
"g"	Green
"b"	Blue
"y"	Yellow
"c"	Cyan
"m"	Magenta
"k"	Black
"w"	White

Sym.	Line Styles
"o" / "s"	Dot, Square
"^", "V", "<", ">"	Triangles
"p"	Pentagon
"*"	Star
"x"	Crossmark
"D"	Diamond
"-", "--", "-.", "."	Solid, Dash-Dash, Dash-Dot, Dot lines

Line Chart: plot()

- **Multiple Sets of Data:**

```
1  import matplotlib.pyplot as plt
2  import numpy as np
3
4  X = np.arange(0., 5., 0.2)
5  plt.plot(X, X, "r--")
6  plt.plot(X, X**2, "bs")
7  plt.plot(X, X**3, "g^")
8  plt.show()
```

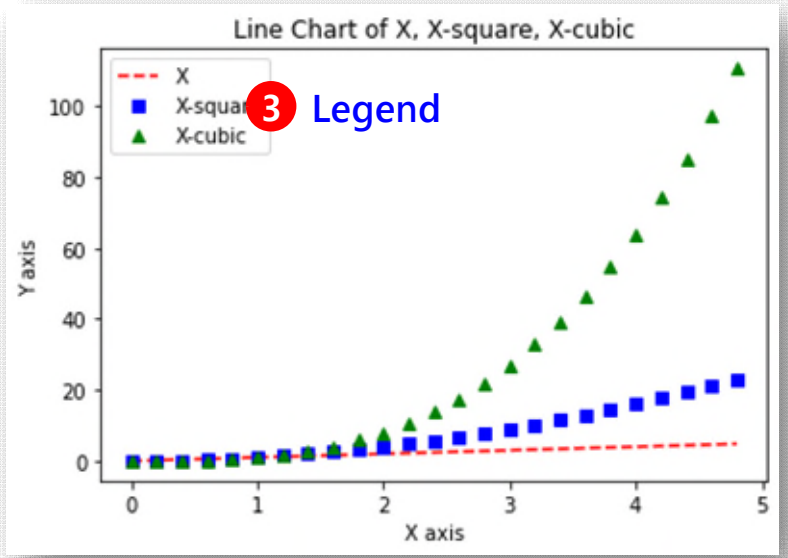


Practice

- **Line Chart: plot()**
- **Import the following packages**
  - import matplotlib.pyplot as plt
  - import numpy as np
- **Create the X-Data:**
  - X = np.*arange*(**0.**, **5.**, **0.2**)
- **Please use X, X2, and X3 as Y-Data to draw the line chart:**
  - plt.**plot**(X, X, **"r--"**)
  - plt.**plot**(X, X**2, **"bs"**)
  - plt.**plot**(X, X**3, **"g^"**)
- **Here is the code for reference:**

```
1  import matplotlib.pyplot as plt
2  import numpy as np
3
4  X = np.arange(0., 5., 0.2)
5  plt.plot(X, X, "r--")
6  plt.plot(X, X**2, "bs")
7  plt.plot(X, X**3, "g^")
8  plt.show()
```

Title, X/Y-labels, and Legend




```python
import matplotlib.pyplot as plt
import numpy as np

X = np.arange(0., 5., 0.2)

plt.title("Line Chart of X, X-square, X-cubic")
plt.xlabel("X axis")
plt.ylabel("Y axis")

plt.plot(X, X, "r--", label="X")
plt.plot(X, X**2, "bs", label="X-square")
plt.plot(X, X**3, "g^", label="X-cubic")
plt.legend(loc='best')

plt.show()
```

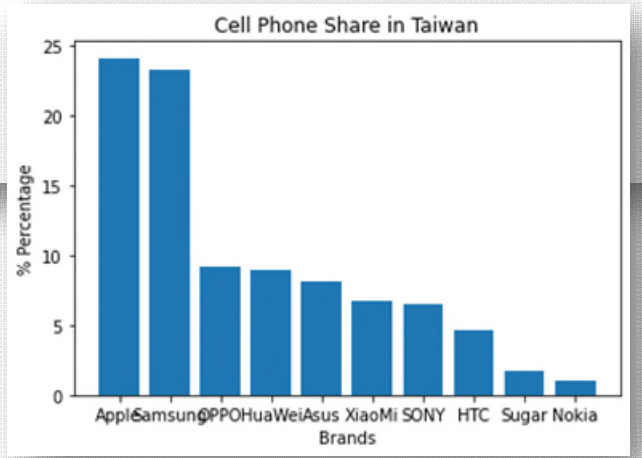
1 2 3

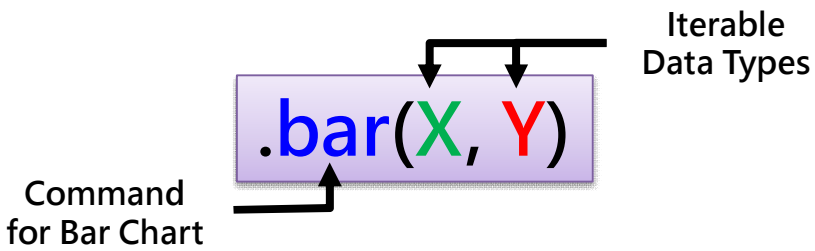
Location (loc=) of Legend :

- "best" , "center" , "upper" , "lower" ...

Bar Chart: bar()

```
1  import matplotlib.pyplot as plt
2
3  X = ["Apple", "Samsung", "OPPO", "HuaWei", "Asus", "XiaoMi", "SONY", "HTC", "Sugar", "Nokia"]
4  Y = [24.1, 23.3, 9.2, 8.9, 8.1, 6.7, 6.5, 4.6, 1.7, 1.0]
5
6  plt.title("Cell Phone Share in Taiwan")
7  plt.xlabel("Brands")
8  plt.ylabel("% Percentage")
9
10 plt.bar(X, Y)
11 plt.show()
```





Practice

- **Bar Chart: bar()**

```
1  import matplotlib.pyplot as plt
2
3  X = ["Apple", "Samsung", "OPPO", "HuaWei", "Asus", "XiaoMi", "SONY", "HTC", "Sugar", "Nokia"]
4  Y = [24.1, 23.3, 9.2, 8.9, 8.1, 6.7, 6.5, 4.6, 1.7, 1.0]
5
6  plt.title("Cell Phone Share in Taiwan")
7  plt.xlabel("Brands")
8  plt.ylabel("% Percentage")
9
10 plt.bar(X, Y)
11 plt.show()
```

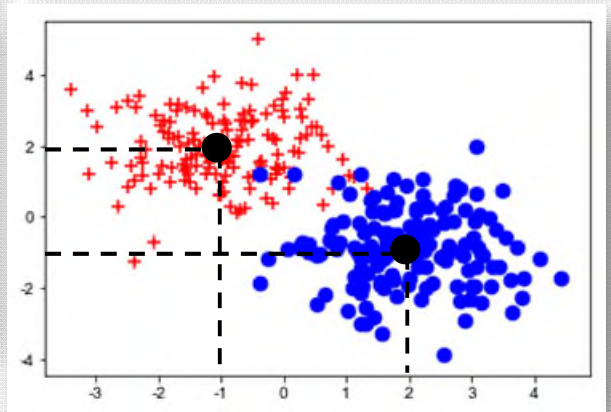
Scatter Chart: scatter()

```python
import matplotlib.pyplot as plt
import numpy as np

# Size of Sample Data
n = 150

# Group1 Data around (-1, 2) as Normal Distribution
X1 = np.random.normal(-1, 1, n)
Y1 = np.random.normal(2, 1, n)

# Group2 Data around (2, -1) as Normal Distribution
X2 = np.random.normal(2, 1, n)
Y2 = np.random.normal(-1, 1, n)

# Scatter Chart with Size(s), Color(c), Style(marker) of marker
plt.scatter(X1, Y1, s=75, c="red", marker="+")
plt.scatter(X2, Y2, s=75, c="blue", marker="o")
plt.show()
```

Center=(-1, 2)
Std=1 x150 Samples

Center=(2, -1)
Std=1 x150 Samples

size=75px, Color=Red/Blue
Marker=Crossmark/Circle



Scatter Chart: scatter()

- **Scatter Chart: scatter()**

```
1  import matplotlib.pyplot as plt
2  import numpy as np
3
4  # Size of Sample Data
5  n = 150
6
7  # Group1 Data around (-1, 2) as Normal Distribution
8  X1 = np.random.normal(-1, 1, n)
9  Y1 = np.random.normal(2, 1, n)
10
11 # Group2 Data around (2, -1) as Normal Distribution
12 X2 = np.random.normal(2, 1, n)
13 Y2 = np.random.normal(-1, 1, n)
14
15 # Scatter Chart with Size(s), Color(c), Style(marker) of marker
16 plt.scatter(X1, Y1, s=75, c="red", marker="+")
17 plt.scatter(X2, Y2, s=75, c="blue", marker="o")
18 plt.show()
```